WebGIS 课程实习报告

学院：	资源与环境科学学院
专业：	地理信息科学
班级：	地信 3 班
姓名：	刘子煜
学号：	2015301110036
教师：	黄丽娜 王明军

目录

一、 项目目标

本项目为 WebGIS 系统开发项目工程。通过给定空间数据并自行组织基础底图，自定义专题信息，利用商业、开源软件或自主开发，实现网络环境下空间信息发布、专题信息在线编辑、查询、统计等功能。课程项目旨在加强 WebGIS、网络编程、空间数据库管理、信息共享与发布等理论学习基础上的实践动手能力，掌握一定相关技术的底层开发方法。

项目最终完成个人地图网页的制作，并将其部署到云服务器，从而可以从外网进行访问。网页包括对空间信息的显示、发布、查询、数据库的管理维护、数据统计分析等功能。

二、 开发环境

开发操作系统：Windows2007　　开发 IDE：MyEclipse 10

语言：java、html、javascript、geojson、json、css

无论何种开发环境，WebGIS 与空间信息共享涉及服务器端部分（Web Server）和客户端（Browser）两大部分，其平台选择也分为服务器端（如：Tomcat、IIS、WebSphere 等）和客户端（IE、FireFox、360 浏览器、Chrome 等）。本系统采用基于 J2EE 架构搭建，

客户端：采用 OpenLayers3：专为 Web GIS 客户端开发提供的 JavaScript 类库包，用于实现标准格式发布的地图数据访问。

服务器端：采用 TomCat7.0 服务器：一个免费的开放源代码的 Web 应用服务器，属于轻量级应用服务器；应用 Geoserver 发布地图数据；阿里云服务器部署网页。

数据库：应用 Postgresql 数据库和 postgis 空间数据库存储空间地理数据和其他关系数据，并通过网络传输数据。

三、 系统设计

(一) 网络结构

系统利用 B/S 结构（Browser/Server），即浏览器/服务器模式。B/S 结构是 WEB 兴起后的一种网络结构模式，WEB 浏览器是客户端最主要的应用软件。这种模式统一了客户端，将系统功能实现的核心部分集中到服务器上，简化了系统的开发、维护和使用。客户机上只要安装一个浏览器，如 Chrome 或 Internet Explorer，服务器安装 PostgreSQL、SQL Server、Oracle、

MYSQL 等数据库。浏览器通过 Web Server（TomCat Server） 同数据库进行数据交互。

B/S 瘦客户端模式，极大简化了客户端，几乎不需要维护。

B/S 架构管理软件只安装在服务器端（Server）上，网络管理人员只需要管理服务器就行了，用户界面主要事务逻辑在服务器（Server）端完全通过 WWW 浏览器实现，极少部分事务逻辑在前端（Browser）实现，所有的客户端只有浏览器，网络管理人员只需要做硬件维护。为减轻用服务器运行数据的重荷，一般备有数据库存储服务器，最终实现客户端、服务器端和数据库服务器三层架构。

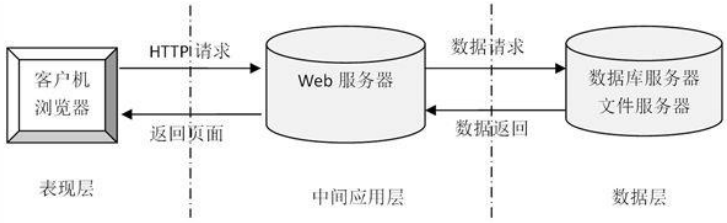

Figure 1　B/S 三层架构图

(二)　逻辑结构

本系统逻辑结构主要分为：数据层、服务层、功能层、用户操作层。数据层为系统底层数据基础，包括地理空间数据、用户信息数据、分类统计数据等；服务层主要是对功能的实现，包括用户管理、数据更新查询、文件传递、数据分析统计等，服务器端的逻辑实现，为功能层的各个功能的实现提供了基础、用户操作层在交互友好界面基础上应用系统各个功能。具体结构如下图：

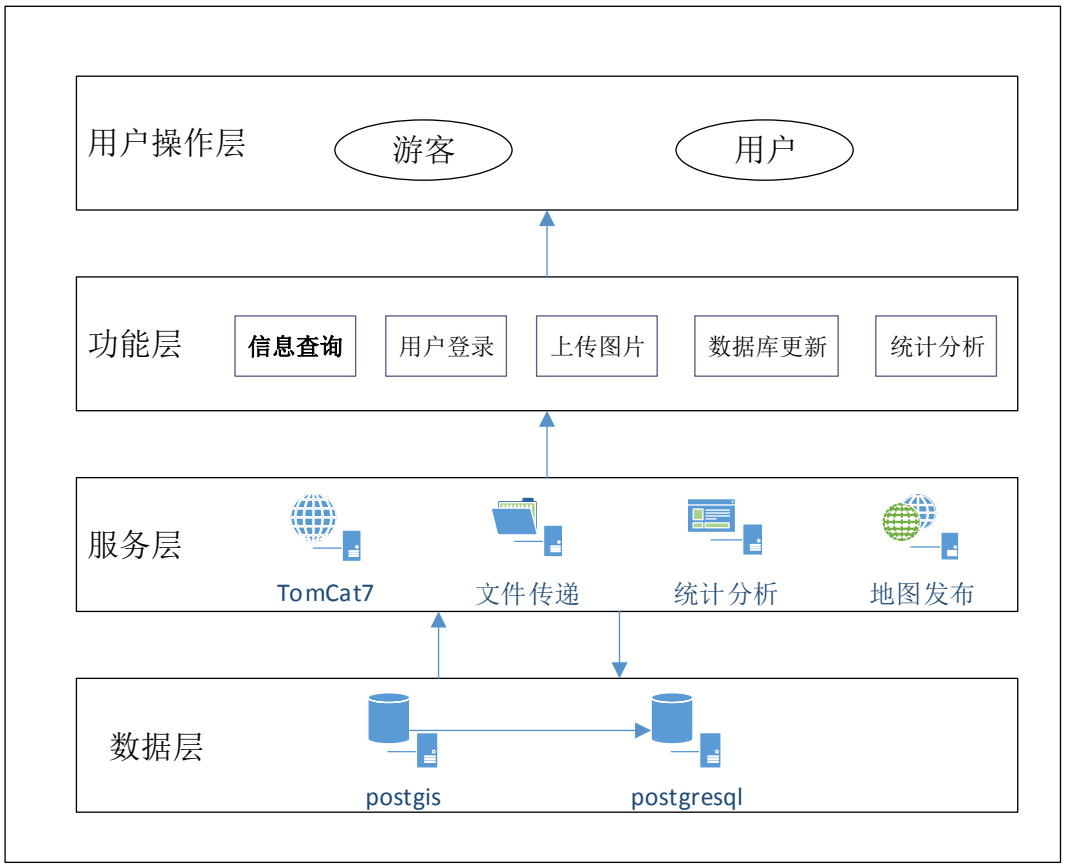

Figure 2 系统逻辑结构图

系统操作逻辑为：首先进入系统主页，包括登录、注册、游览等操作按钮，可先进行用户登录注册等功能，也可以在非登陆的状态下游览地图，进行简单的空间和属性查询。登陆后页面跳转到地图主页，可以进行空间查询、属性查询（显示城市基本属性和天气状况以及城市图片等介绍信息）、文件的上传和数据库的更新、以及实时空气质量统计图表的查看。

(三) 数据结构

本系统中数据主要分为空间数据和用户信息登记、日志管理数据。

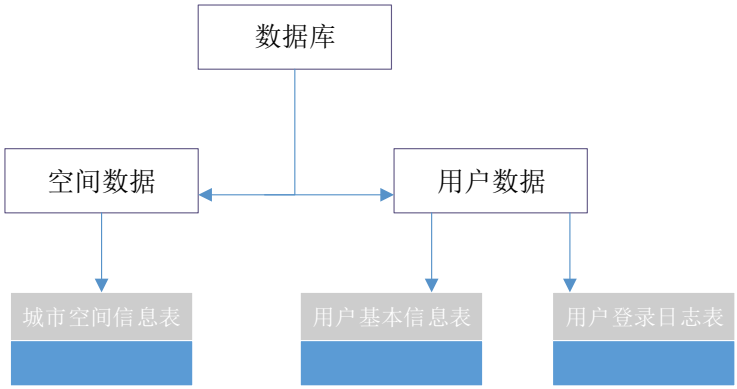

Figure 3 系统数据结构图

空间数据包括空间坐标数据、几何数据和属性数据。本系统使用全国主要城市数据，具体字段有：name、pinyin、gid、adclass、adcode、geom，后又自主添加字段 image、AQI、airlevel 进行数据库的增添删改操作。用户信息登记数据包括用户 id、用户名称和密码；日志管理数据则为用户登录退出记录表，包括用户名称登陆时间、退出时间、登录时长等。

(四)　功能结构

系统功能分为两个不同类型：游客型和用户型。游客只能游览地图并进行简单的空间查询和属性查询，用户则拥有更高的权限，可以进行复杂的查询操作，并上传图片删改数据库内容，获取实时空气质量数据，进行数据库更新和数据的统计分析。具体结构如下图：

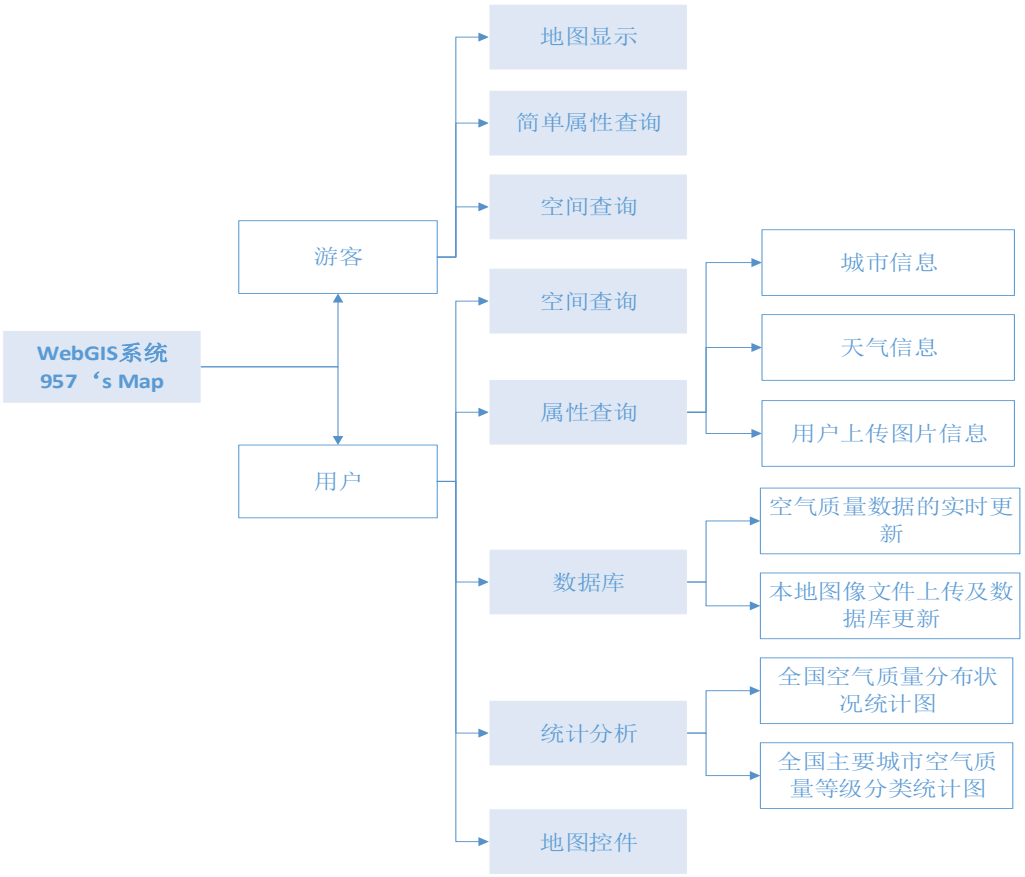

Figure 4　系统功能结构图

四、　环境搭建与数据准备

(一)环境搭建

1. Java jdk

安装 Java 虚拟机（JVM），建立 Web Server（Tomcat）与 GIS/Map Server 的基础库。

Java 虚拟机（Java Virtual Machine 简称 JVM）是运行所有 Java 程序的抽象计算机，是 Java 语言的运行环境，Java 虚拟机屏蔽了与具体操作系统平台相关的信息，使得 Java 程序只需生成在 Java 虚拟机上运行的目标代码（字节码），就可以在多种平台上不加修改地运行。

Java jre 是 tomcat 服务器的运行环境，Java jdk 提供编译 Web 工程的工具，所以在安装 TomCat 之前需要在计算机上安装 Java，并配置其环境。

具体方法：

1）在官网下载 Java 安装包：http://www.oracle.com/technetwork/java/javase/ downloads/jdk8-downloads-2133151.html，选择版本 1.8（Geoserver 不支持 JDK 9 及其以上版本），注意 Java 版本对计算机系统的选择，本项目选择 32 位安装包。 默认安装。

2）安装之后要对环境变量进行配置。找到安装路径：C:\Program Files\Java\jdk1.8.0_131，在计算机属性->高级系统设置->环境变量->系统变量->新建，新建 JAVA_HOME，变量值为 jdk 的文件路径，CLASSPATH 变量，变量值%JAVA_HOME%\lib;%JAVA_HOME%\lib\tools.jar，并将新建的两个变量添加到 Path 中，确定保存，计算机的 Java 运行环境即配置完成。

3）通过 java -version 检测 java 的版本，确定 java 的安装。

2. TomCat7.0

安装开源 Tomcat 作为 Web Server。

Tomcat 是 Apache 软件基金会（Apache Software Foundation）的 Jakarta 项目中的一个核心项目，由 Apache、Sun 和其他一些公司及个人共同开发而成。Tomcat 5 支持最新的 Servlet 2.4 和 JSP 2.0 规范，技术先进、性能稳定，而且免费，是目前比较流行的 Web 应用服务器。

Tomcat 服务器是一个免费的开放源代码的 Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试 JSP 程序的首选。当在一台机器上配置好 Apache 服务器，可利用它响应 HTML（标准通用标记语言下的一个应用）页面的访问请求。故而在进行 Web 工程开发之前，首先要搭建好 TomCat 服务器。

具体方法：

1）在官网 http://tomcat.apache.org/下载 TomCat 压缩包。选择版本 7。

2）将压缩包解压到英文安装目录下，配置 TomCat 的环境变量，设置系统变量 CATALINA_HOME，值为 TomCat 的安装路径，并将该变量加入到 Path 变量中。

3）利用命令控制台打开 bin 目录里的 startup.bat 文件，运行 TomCat，运行 shutdown.bat，停止 TomCat。

4）将 TomCat 部署到云服务器端还需更改端口号。打开安装目录下 conf 文件夹中的 server.xml 文件，找到如下代码，将 8080 端口改为 80（更改之前可以用命令行判断 80 端口是

否被占用）。

```
<Connector port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

Figure 5 TomCat 端口更改代码显示图

5）启动 TomCat，打开地址 http://locathost:8080/（8080 端口号注意同修改的端口号一致），若出现下图界面，则部署成功。

注：通过云服务器公网如：http://39.108.133.182/ 登录 TomCat 页面时，需要设置服务器端的安全组规则。具体规则如下图：

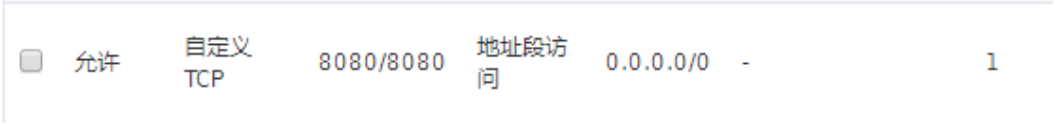

Figure 6 云服务器安全组规则设定图

6）在 conf 文件夹中有 tomcat-users.xml 文件，通过修改用户名和密码，重启 TomCat 获得账号登录其管理状态。

```
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
```

Figure 7 TomCat 用户设置图

3. GeoServer

安装 GeoServer 地图服务软件环境。GeoServer 是 OpenGIS Web 服务器规范的 J2EE 实现，利用 GeoServer 可以方便的发布地图数据，允许用户对特征数据进行更新、删除、插入操作，通过 GeoServer 可以比较容易的在用户之间迅速共享空间地理信息。

GeoServer 兼容 WMS 和 WFS 特性；支持 PostgreSQL、Shapefile、ArcSDE、Oracle、VPF 、MySQL 、 MapInfo 等数据库和数据格式；并嵌入有 MapBuilder 支持 AJAX 的地图客户端 OpenLayers；除此之外还包括许多其他的特性。

安装及地图发布方法：

1）在官网 http://geoserver.org/release/2.8.2/ 下载开源 GeoServer 的 war 压缩包（注意对应版本），拷贝到 Tomcat 所在的 webapp 目录中。

2）启动 TomCat，出现如下窗口：

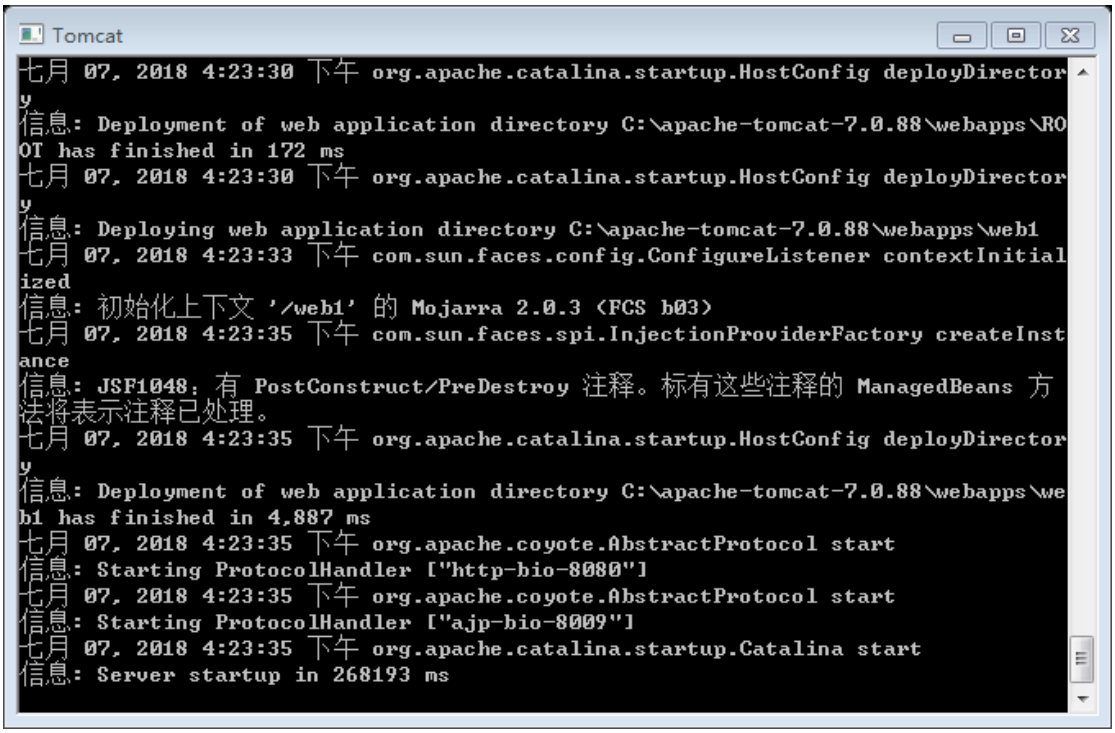

Figure 8 TomCat 成功启动示意图

3）在浏览器中配置 geoserver，打开网页 http://localhost:8080/geoserver 则 geoserver 安装成功。

4）发布 shp 地图：登录身份，在侧边栏数据菜单中选择工作区，创建自己的工作空间和地址，并命名空间。

Figure 9 geoserver 新建工作区示意图

5）选择数据存储->添加新的数据存储->选择矢量数据源 Shapefile，在自己的工作区中导入本地 shp 文件，设定投影坐标系计算图层边界后发布。

6）在 Layer Preview 中找到自己发布的图层，通过 OpenLayers 的方式查看，如下图：

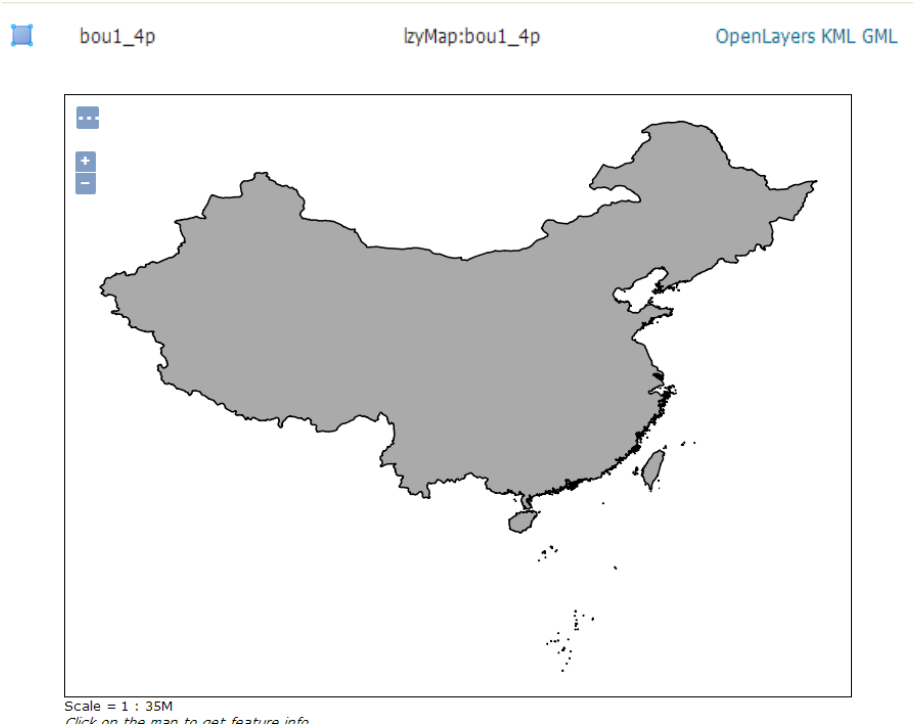

Figure 10 geoserver **地图发布示意图**

注意地图发布的网址参数：

http://localhost:8080/geoserver/lzyMap/wms?service=WMS&version=1.1.0&request=GetMap&layers=lzyMap:bou1_4p&styles=&bbox=73.44696044921875,6.318641185760498,135.08583068847656,53.557926177978516&width=768&height=588&srs=EPSG:4326&format=application/openlayers

4. MyEclipse

开发环境安装，利用 MyEclipse 环境实现服务器端功能开发和客户端页面开发。

MyEclipse 企业级工作平台（MyEclipseEnterprise Workbench ，简称 MyEclipse）是对 EclipseIDE 的扩展，利用它我们可以在数据库和 JavaEE 的开发、发布以及应用程序服务器的整合方面极大的提高工作效率。它是功能丰富的 JavaEE 集成开发环境，包括了完备的编码、调试、测试和发布功能，完整支持 HTML，Struts，JSP，CSS，Javascript，Spring，SQL，Hibernate。

环境配置：将 MyEclipse 和计算机上的 Java JDK、TomCat 关联，配置开发环境的编译环境和 src，使得新建的 Web 工程可以搭建在 TomCat 服务器之上。

(二)数据准备

1. 数据库安装

系统应用 PostgreSQL+PostGIS 空间数据库管理数据。

在官网https://www.enterprisedb.com/downloads/postgres-postgresql-downloads#windows 下

载PostgreSQL（9.6）安装软件和PostGIS插件（与PostgreSQL版本一致），按照安装向导进行安装。在安装PostGIS时注意勾选Create SpatialDataset，注意更改数据库名称，解决空间数据库建立失败的问题。记录安装数据库时填写的账号、密码和数据库名称。

2. 数据导入

打开PostGIS 2.0 ShapFile and DBF Loader Exporter，登录数据库账号密码和数据库名称，并在Options菜单栏中设定数据库编码格式和geometry格式，选择本地shp文件import。注意更改shp文件投影坐标系。

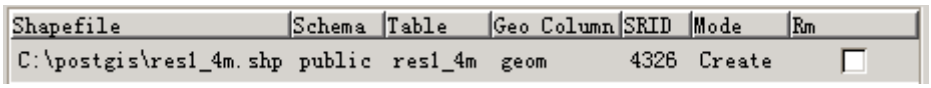

Figure 11 postgis 数据导入示意图

对话框显示如下图即导入成功。

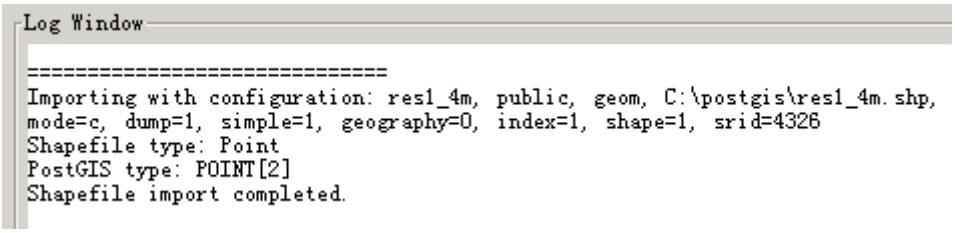

Figure 12 数据导入空间数据库成功示意图

五、 功能模块实现

本WebGIS项目工程主要由3个显示jsp页面、4个隐式jsp页面、3个java类文件、1个css文件和部分布局设计图片组成。其间结构图如下：

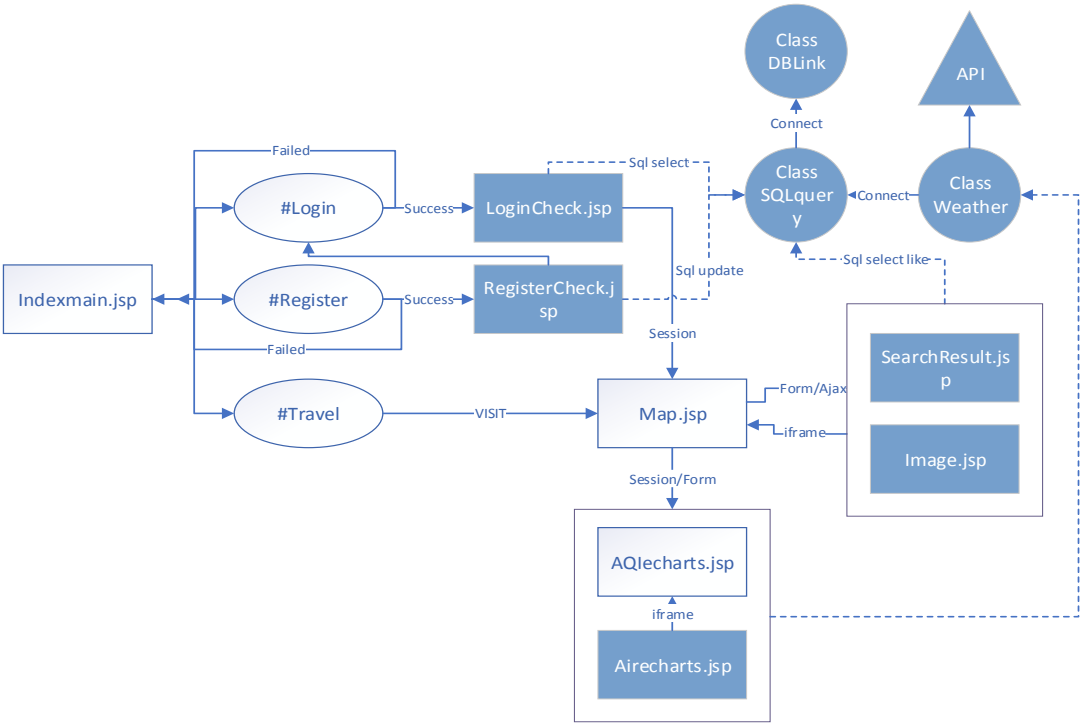

Figure 13 Web 工程文件消息响应结构图

主页面indexmain.jsp为项目欢迎界面分别连接Login登录、Register注册和Map地图游览功能，登录和注册界面借用隐式界面进行服务器端与数据库之间的操作。登录后，Map.jsp集合了，查询、数据库编辑、图片上传等功能控件。在查询功能、图片上传功能中利用form表单提交传递查询参数，子页面在服务器端调用java类进行分析获得结果，由iframe标签获取跳转页面显示在父页面上。统计图表则跳转单独页面AQIecharts.jsp进行显示。

1. 数据库连接

数据库在 web 工程中通过后台服务器端进行连接操作。在使用数据库之前首先要连接到本地数据库。

首先在 MyEclipse Web 工程中导入 postgresql 库到 lib 文件目录下，右击项目目录->Build Path->Configure->Add JARs 将 postgresql 数据库加入编译路径之中。

新建 Java 类文件 DBLink，引用库文件中的 Connection、DriverManager、SQLException 类。新建 Connection 类通过 DriverManager 类依据数据库用户名、密码和空间数据库表连接数据库。Connection 类非空则数据库连接成功。

2. 地图显示：

新建 jsp 动态网页 Map.jsp，在页面添加 div 标签作为地图容器。在页面 js 中定义初始化函数 init()，并在页面 body 标签加载时对页面进行初始化设定。在 Init 函数中实现地图的显示。地图显示主要主要基于 OpenLayers3 库，分为 3 部分：底图 osm 显示、wms 地图显示、和数

据库空间地理信息数据读取 geojson 格式显示。

1) 添加 wms 图层及 OSM 地图

利用 Openlayers3 提供的 ol.layer.tile 接口将已发布在 geoserver 上的 wms 图层添加到 layers 变量中，并通过 ol.source.TileWMS 接口指定 geoserver 中 wms 服务地址及服务名称作为图层数据源，实现 wms 地图的显示；利用 ol.source.OSM()接口新建 osm 地图数据，作为数据源，将 osm 瓦片地图同样添加到 layers 变量中。

2) 从数据库添加图层到网页

从数据库获取 geojson 文件：新建 Java Class，SQLquery 类实现数据库 sql 语言查询。利用“select *,ST_AsGeoJson(geom) from res1_4m”语句获取 shp 文件中的 geom 属性，并编写相关属性，形成该 shp 文件对应的完整的 geojson 文件。在 Map.jsp 初始化函数中应用后台 Java 语言调用 SQLquery 类获取 geojson 数据。

3) 添加矢量图层

以 geojson 的数据格式从数据库获取空间矢量数据（shp 文件），作为矢量图层的数据源，显示在地图页面上。

调用 openlayers3 的 ol.layer.Vector 矢量图层类来绘制从 geojson 文件。利用 ol.format.GeoJSON()类从 geojson 中读取要素作为矢量源 ol.source.Vector 的要素，并将该矢量源作为矢量图层源，创建图层对象，并添加到 init()函数的 layers 变量中。

4) 地图在页面上的显示

利用 openlayers3 的 ol.Map 接口，新建集合上述图层对象的 map 对象变量，并利用 ol.View 接口设置图层投影方式 EPSG:4326；显示中心[116.5, 39.5]和显示层级（地图缩放等级）：4。并将地图变量作为 map div 的内容显示在页面。

5) 实现成果如图：

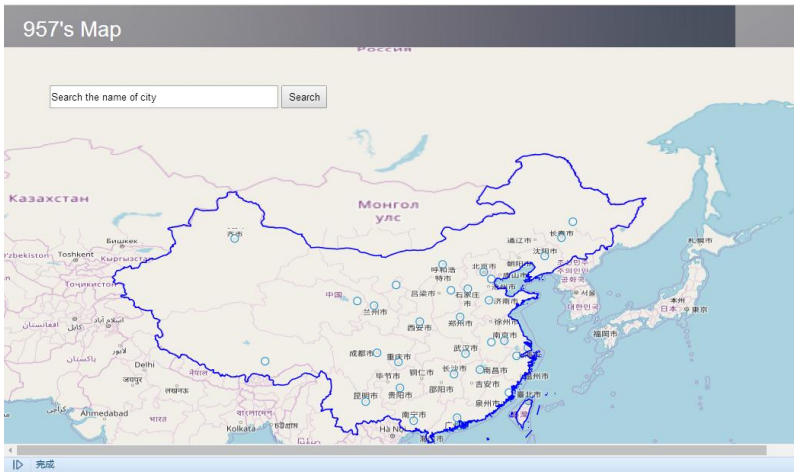

Figure 14 图层显示成果图

3. 地图查询

1) 属性查询

输入城市名称进行查询，在网页侧边栏显示城市相关信息。

属性查询主要使用 form 表单，action 跳转新的 SearchResult.jsp 文件并传递客户端查询的城市名称的参数，jsp 刷新运行后端服务器代码，在 SQLquery 类中建立查询函数，用 sql 语句在数据库中获取目标城市相关信息，并将信息显示在客户端 SearchResult.jsp 中。设置 form 表单参数 target，使其指向 iframe 标签，iframe 作为子页面容器，可以在 form 表提交时不用跳转页面，不刷新父页面，直接加载子页面实现查询信息的反馈。

Map.jsp form/iframe SearchResult.jsp(request.getparameter) 后台 SQLquery sql DBLink 前端 SR.jsp iframe

在属性查询中，有一下几点需要特别注意：

a）模糊查询。用户输入的查询关键字存在拼音、汉字和拼写错误等情况，故而利用 sql 语句编辑进行模糊查询。需将汉字转换成拼音，利用 pinyin4j.jar 包，判断获取的城市姓名参数是否为汉字字符，若是汉字则利用 PinyinHelper.toHanyuPinyinStringArray()接口进行转换。在 SQLquery 类的查询函数的编写中，将 sql 语句中的“=”改为 like，使得相似的城市名称参数都可以找到对应的结果。

b）用户权限。在游览状态下的用户无法获取全部的应用结果，故而在 SearchResult.jsp 界面中，利用 session 对是否是登录用户进行判断，若是则将 css 中利用“display:none”隐藏的 div 标签显示出来，体现用户与非用户的差别。

2) 空间查询

点集地图上的城市点，在漂浮窗口上显示城市的名称和坐标信息。

在 init()函数利用 Openlayers3 的 ol.interaction.Select()接口创建 click 交互方式赋值给 select，并将该交互添加到 map 变量中，对 select 变量添加.on 监听事件，从矢量图层 geojson 中获取点坐标和城市名称等信息。

新建 popup 弹出框并 id 命名，将其 innerHTML 设置为查询到的城市信息，并设计 popup 的 css 显示布局格式等，利用 ol.Overlay 将弹出框作为元素添加到 map 地图上，并利用城市坐标信息设置 overlay 的位置，使得弹出框出现在目标城市上方。

3) 成果显示如下：

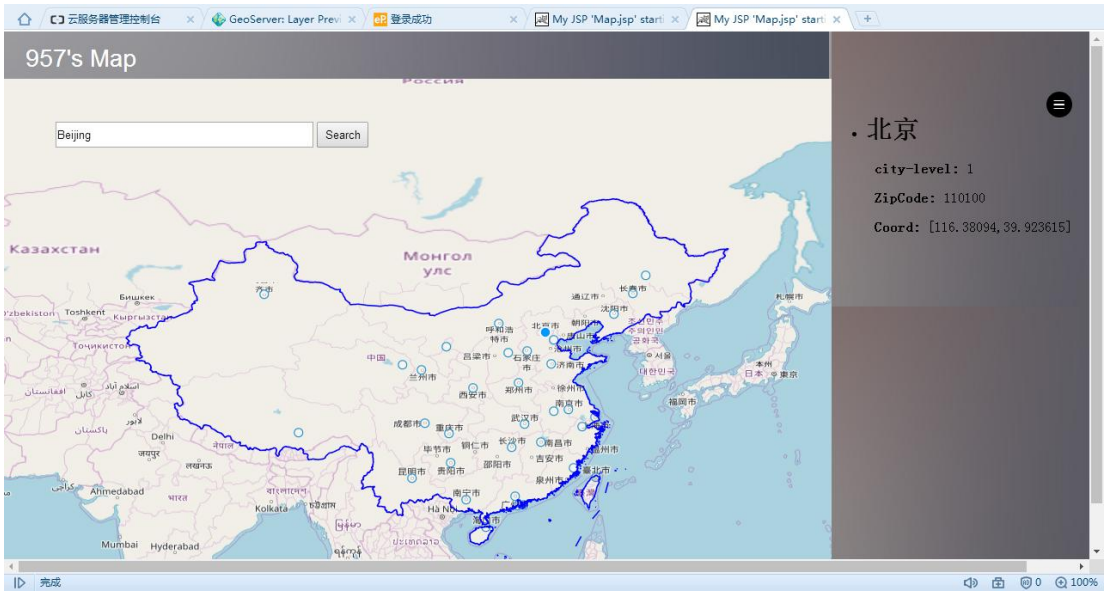

Figure 15 非登录用户游览模式查询结果图

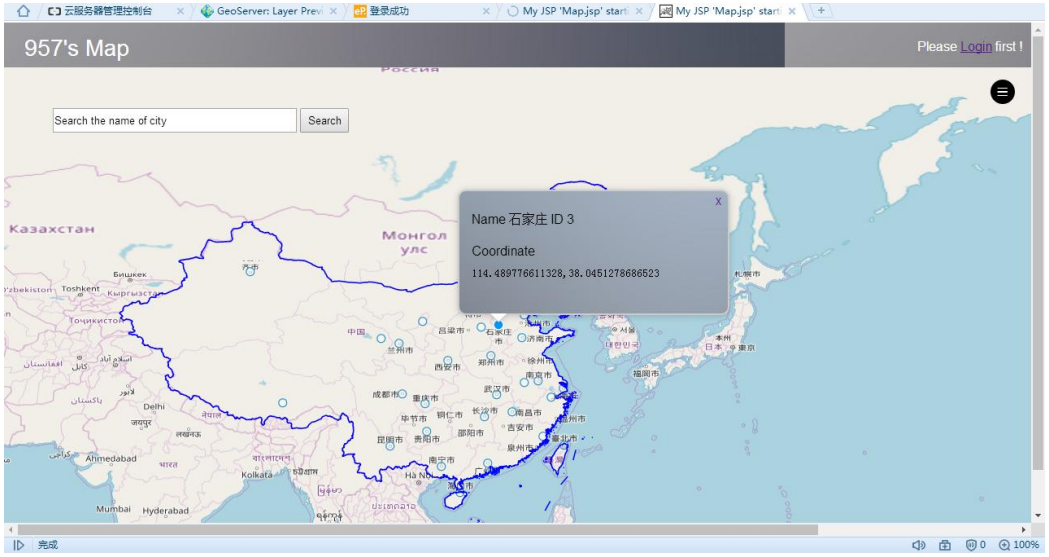

Figure 16 空间查询结果图

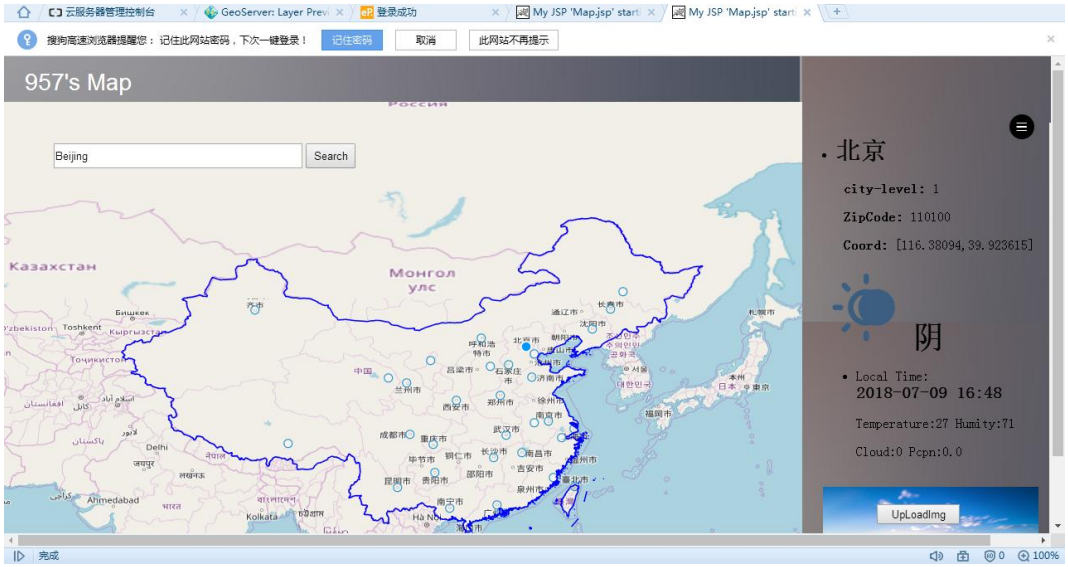

Figure 17 登录用户属性查询结果图

4. 图片上传与显示

1) 图片上传到数据库

利用 form 表单和 type 为 file 的 input 标签将图片数据和图片对应城市名称传递到 Image.jsp 页面中，文件传输方式选择 enctype ="multipart/form-data"，将图片二进制流数据完整打包传递到新页面之中。若默认文件传输方式，新页面 request.getParameter()无法获取文件数据，获得参数值为空。未获得已上传的二进制流数据，添加 SmartUpload.jar 包对打包数据进行分析解析。通过新建 SmartUpload 类对象，对该对象进行初始化上传操作、上传准备，通过 SmartUpload.getRequest().getParameters（）获取城市名称参数。DBLink 连接数据库，新建数据库查询声明和结果，调用 SmartUpload.getFiles().getFile(0).fileToField()函数将二进制流图片数据添加到对应城市数据的“image”字段中，更新结果字段。二进制流上传到数据库完成。

2) 从数据库获取图片显示

利用将图片二进制流数据转化为 base64 数据格式。新建 img 标签，设置 src 为“”+图片 base64 码，即可显示在页面。

3) 成果显示如下：

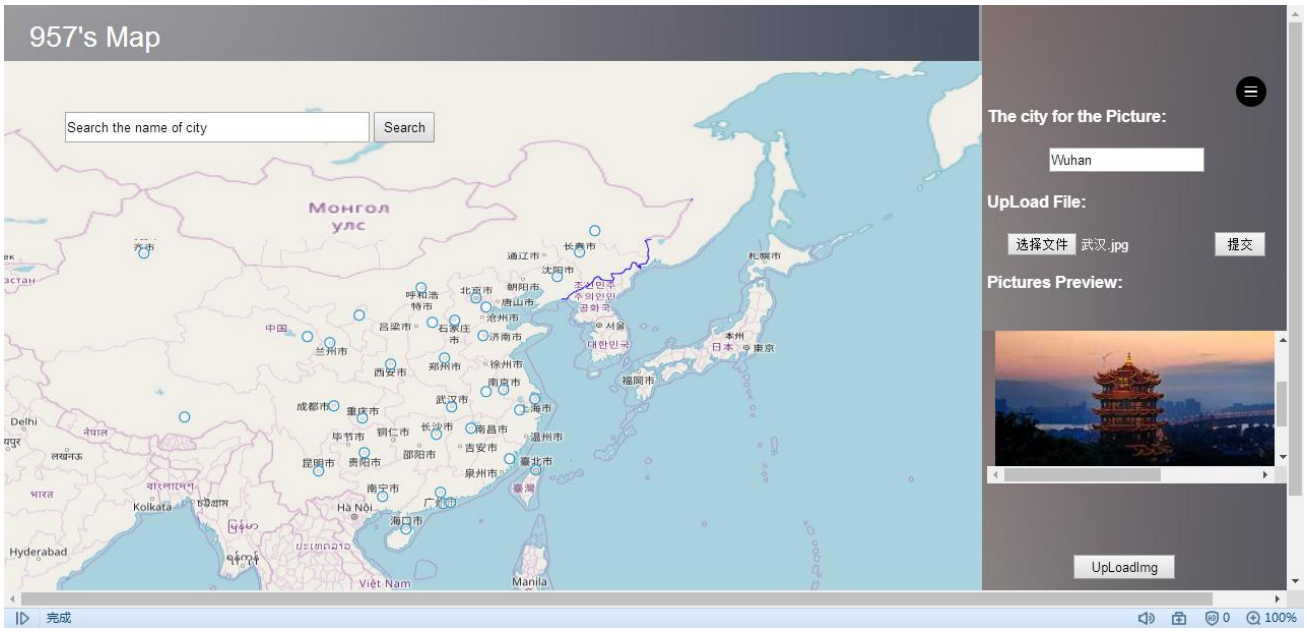

Figure 18 **图片上传成果展示图**

5. 用户登录及注册

1) 用户登录注册设计

在数据库中新建表 users，添加用户名和密码两个字段存储用户注册信息。

设计用户注册客户端界面和后台服务器程序。利用<table>标签规整注册信息填写表格，并设置输入框文本格式。利用标签属性设置输入框输入字符的长度限制，reset 型 input 控件实现输入框的一键重置。提交按钮首先触发点击函数，确认输入信息是否符合规则，即两次输入密码结果是否一致，不一致则弹出消息框并清空注册表，重新输入注册信息；若信息检查合格则跳转 RegisterCheck.jsp 注册确认页面，在启动页面时，运行后台服务器程序。连接数据库，查询数据库用户是否存在，若存在则跳转回注册页面，不存在则在数据库中 Insert 新的数据记录，并跳转到登陆页面，用户注册成功。

设计用户登录界面，利用 table 标签将用户名、密码及其对应的输入框规整排列，input 标签 submit 控件作为登录按钮跳转登录检验 LoginCh.jsp 进行登录验证。在数据库中 select 用户和密码比对之后，登陆成功则随即跳转 Map.jsp 页面。登录失败则返回登录页面。

2) 用户登录日志管理

利用session在服务器端记录用户的登录和退出状态和时间。新建User java类，记录用户名称和密码，在LoginCh.jsp页面服务器端处理代码中通过 request.getSession().setAttribute("user", user)将用户信息添加到session，并将session的添加时间和user相关信息insert到用户登录日志表中，进行记录。在Map.jsp页面，通过 request.getSession(false); User) session.getAttribute("user");获取session中记录的user对象，根据

user是否为空来判断界面是否未登录状态，并选择性显示相关div。

用户的退出通过点击lougout <a href>连接跳转到LogoutCh.jsp退出登录确认界面，通过session.removeAttribute("user")接口删去user记录并连接数据库用sql语言update数据库，记录用户退出登录的时间。response.sendRedirect()接口返回登录界面。

登录界面成果图

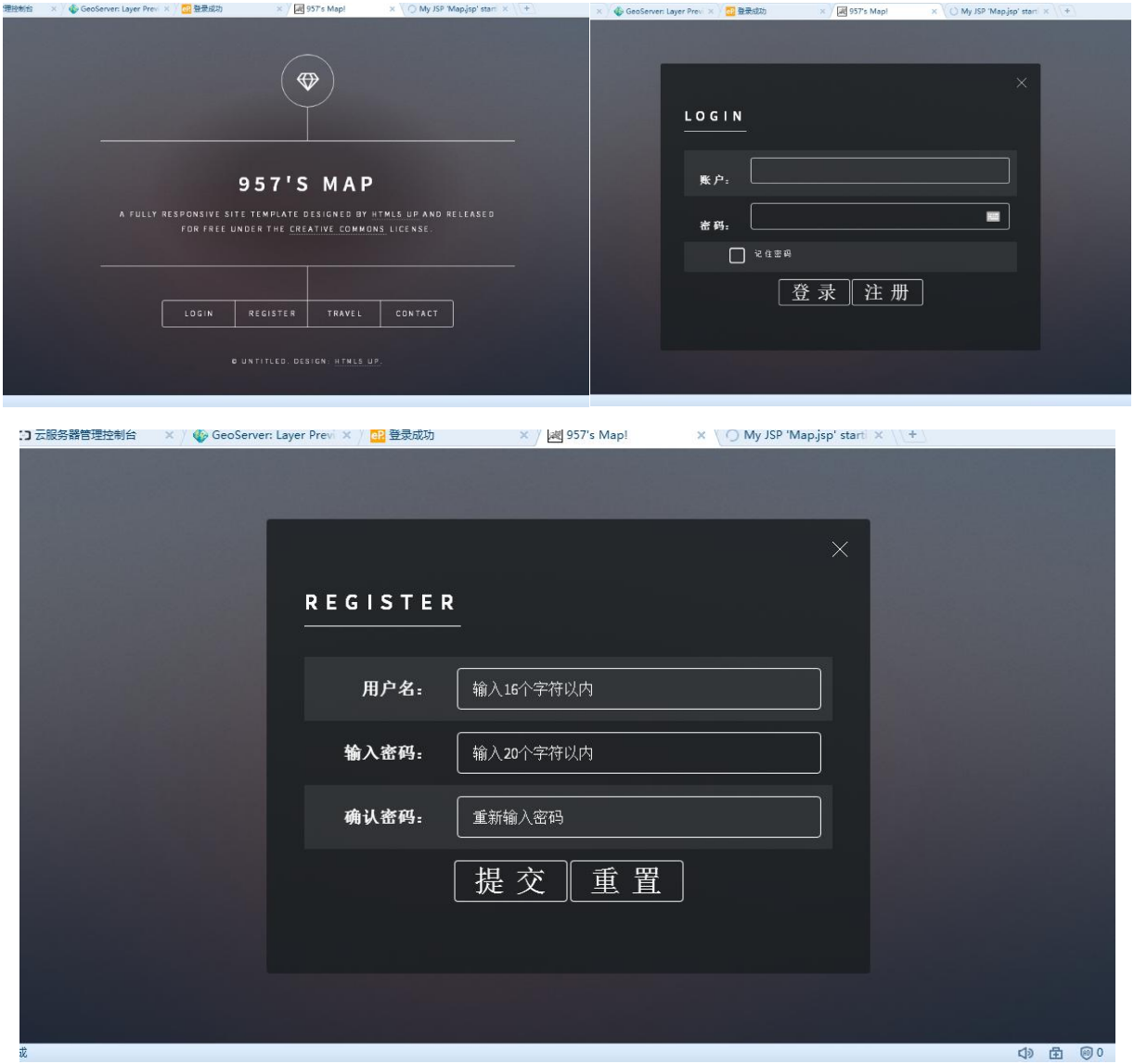

Figure 19 **系统登录界面成果图**

6. 天气、空气质量 API

利用和风天气 API 获取城市实时天气状况。新建 Weather Java 类，利用 HttpURLConnection 类连接 api 的 url，将查询表单提交的城市名称参数和已申请的 API key 作为参数发送到连接中，获取对应城市天气和空气质量的 json 文件。天气 API 为 "https://free-

api.heweather.com/s6/weather/now?";空气质量 API 为 https://free-api.heweather.com/s6/air/now?。

在项目工程中添加 commons-beanutils-1.7.jar、commons-collections.jar、commons-lang.jar、commons-logging-1.1.1.jar、ezmorph.jar、json-lib-2.2.2-jdk15.jar 包，使用 net.sf.json.*库解析 json 文件，获取感兴趣信息，并将天气信息显示在 SearchResult.jsp 中，将该部分 div 隐藏，当用户登录时再显示。空气质量数据获取同样从 API 获取的 json 文件中提取。

结果图在属性查询中有显示。

7. 统计图表

1) 数据准备：

遍历数据库中的城市信息，依次调用空气质量 API 接口，获取每个城市的实时 AQI 指数和空气质量评定等级，并在数据库城市信息表中添加 AQI 和空气质量等级字段，利用 net.sf.json.*库解析 json 文件，获取相应数据，sql 语句"UPDATE res1_4m SET AQI=? , airlevel=? WHERE pinyin='"+Name+"'";更新数据库。数据库定时更新。

2) 全国主要城市空气质量等级分布饼图

在数据库查询类 SQLquery 类中利用 sql 语句“SELECT airlevel,COUNT(airlevel) FROM res1_4m GROUP BY airlevel”对城市空气质量等级进行分组统计，包括“优、良、轻度污染、中度污染和重度污染”，按照统计数据格式编写数据。设置 echarts 图表的属性 option，包括 title、tooltip、legend 和 series，在页面中显示。

3) 全国主要城市 AQI 分布点图

从数据库获取实时 AQI 编写统计图表数据格式，数据分为 AQI 指数数据和城市位置数据，数据通过城市 name 字段一一对应关联起来。

新建动态网页 AQIecharts.jsp 后台服务器端调用 SQLquery 类，向数据库发送请求获取实时 AQI 指数数据，利用 echarts 库，将 AQI 数据带入图表之中，设置图表名称、兴趣点格式，将数据在地图底图上显示。

将饼图作为分布散点图页面的一部分，利用 iframe 标签内容框显示饼图页面。

4) 成果如下：

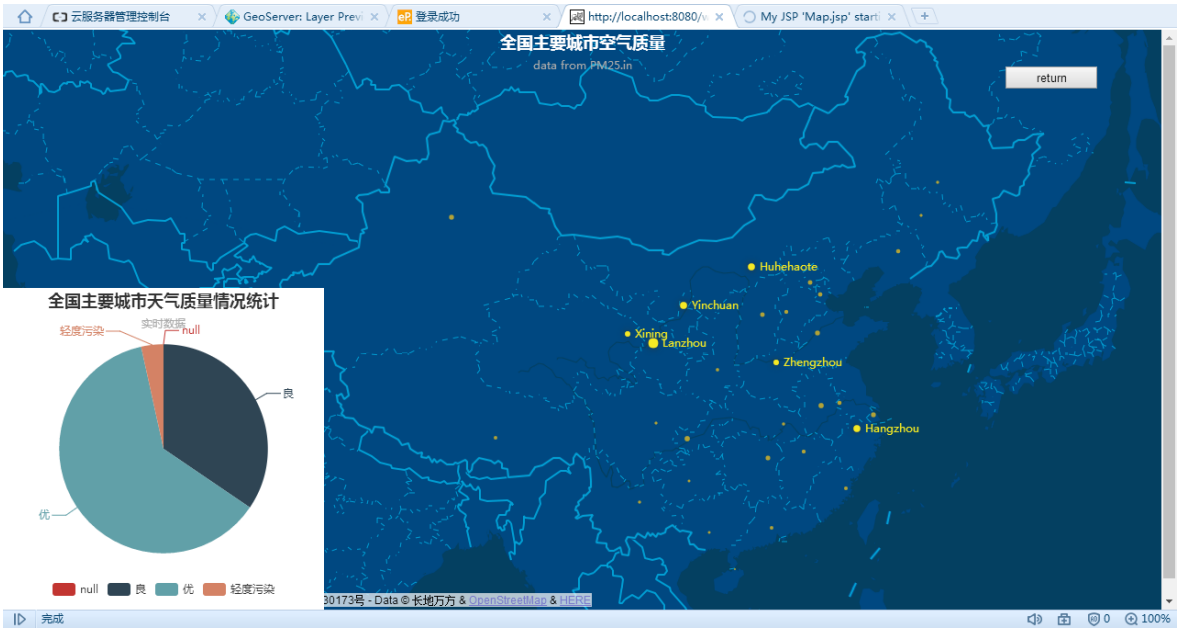

Figure 20 统计图表结果图

六、 实习问题：

(一)环境配置及软件安装

1、问题：配置云服务器 Web 服务器环境时，下载了 Java JDK 9 版本，安装完 TomCat 之后，Geoserver 只支持 Java SDk 8 版本，需要卸载版本 9.而云主机的镜像盘系统为 Ubuntu，由于不熟悉命令行，恍惚中使用了 rm -rf 将云服务器内容全部删除导致 ssh 无法连接到云服务器，port rejected。

解决：理论方案：对云服务器的端口进行配置，开放端口，连接许可。实际方案：初始化磁盘，更改镜像为 windows 系统，利用远程桌面连接登录云服务器，可以进行文件管理等操作。

2、问题：Mac OS 下 MyEclipse 安装 10.7 版本激活失败。根据网上教程对 MyEclipse 进行破解，利用命令行打开 jar 包 java 文件，生成激活码和通行证书，并更改 MyEclipse 安装目录下的注册文件，启动 MyEclipse。结果因安装版本与解压程序版本不符，激活失败。

解决方案：更改系统，使用老师给的 MyEclipse 安装文件。

(二)数据库安装

3、问题：数据库 Postgresql 安装失败。数据库无法初始化。利用.exe 文件安装数据库失败，更改语言配置也没有成功，安装目录下 data 文件夹无法编辑。

解决：将 Postgresql 安装文件夹直接放在 C 盘目录下，而不在 Program(X86)目录下，因为

该目录文件的更改需要特殊权限，一般无法编辑。利用命令行 initdb -D 安装目录\data -E UTF8；pg_ctl register -D 安装目录\data -Npgsql 注册服务，net start pgsql 启动服务，createdb -E UTF8 testdb 创建数据库。残留问题，数据库无法用 pdAmin4 打开。

4、问题：postgis 安装失败，无法创建空间数据库。

解决：更改数据库名称，取消默认。

(三)工程编写

5、问题：网页无法显示创建的 Openlayers 图层。

解决：OpenLayers 库的 src 引入格式错误，jsp 没有在指定目录下找到 OpenLayers3.js 文件。使用网页库链接 http://www.openlayers.org/api/OpenLayers.js 作为 openlayers 库文件。

6、问题：servelet 文件创建失败。原因：Java JDK1,8 版本过高，与 MyEclipse（较低版本）创建的 servelet 版本冲突，新建 servelet 文件失败，无法编译，package 报错。

解决尝试：更改工程编译 JDK 版本为 MyEclipse 自带 Java JDk 1.6，原 Java Class 报错，高版本编译类无法运行，程序无法识别导入的数据库包。

最终解决方案：将 ajax 传递参数方式更改为 Form 表单传递。

7、问题：空间属性查询得到的要素为空。Geojson 文件编写格式只包含了坐标点，要素在绘制到地图上便失去了其地理位置坐标，需要通过要素属性确定要素身份。

解决：重新编写 geojson 文件，对要素添加身份辨别属性，从鼠标点集事件中获取 target.feature，从 feature 中获取 geometry，获得点选要素相关信息。利用 console.log(feature)查看 feature 属性从而调用显示于页面。

8、问题：图片上传

尝试一：form 表单普通文件传输模式，无法获得文件路径，读取图片二进制流数据。IE 浏览器 request 可以获得图片本地路径，Chrome 浏览器只能显示文件名称。

尝试二：利用 enctype="multipart/form-data" 数据传输格式，打包传递文件数据。Request.getparameter 值为 null

解决：引入 SmartUpLoad.jar 包，通过外部封装好的包提取参数 SmartUpLoad.Request().Request.getparameter()；通过外部包获取上传文件，现将文件上传到服务器端，再利用服务器路径获取文件流，写入数据库。问题：服务器端的文件夹会在更新时自动删除，从而无法找到路径，带来问题。

最终解决方案：直接从 Form 表单上传数据中提取文件数据流，写入数据库。利用接口 SmartUpload.getFiles().getFile(0).fileToField()。

9、图片显示问题：二进制流数据从数据库中读取之后用 byte 字节存储 InputStream 通过

rs.getBinaryStream("image")方法读取的数据流，图片输出流 ServletOutputStream 直接在屏幕 out 渲染图片。但图片会以二进制码显现，解决方法更改 response 方式为 image。仍存在问题，无法控制图像大小。

解决方法：转换二进制流为 base64 码，通过 img 标签输出。

10、页面跳转问题：Form 表单提交，action 跳转页面，通过页面的刷新实现参数的传递，而为了页面的简洁不需要页面跳转。

解决：使用 iframe 框架容纳子页面。在 form 中设置属性 target 转到 iframe。即可在父页面看到子页面信息，而父页面没有刷新。

11、问题：云服务器无法用公网 ip 连接，tomcat 更改端口为 80 后依旧无法利用公网 ip 进行连接。

解决方法：在阿里云云服务器端设置入方向端口安全组规则，端口范围 8080/8080，不限制授权对象。

七、 实习创新：

1. **登录界面的创新：**

登录界面 UI 设计别出心裁，并设计游览模式，从工程主页面无需登录直接跳转到参观地图页面。

2. **统计图表的创新：**

实时获取城市空气质量数据绘图，并将空气质量分布图和全国的分级统计图相结合。两个图表相辅相成，实现对全国空气质量实时状况的全方位分析理解。

八、 实习总结：

为期一个月的实习落下帷幕，通过本次实习，基本理解了网络地图的开发，对 WebGIS 有了更加深入的理解。从服务端 Web 服务器的搭建；网络地图数据的发布；网络开发编程的基础知识，Java 语言、js 的编写，html、jsp 静态、动态网页的区别，geojson、json 语言的解析,css 层叠样式语言的编写和实现等；网络消息的提交获取机制，发送 Ajax 请求与 Form 表单请求的区别和联系；网络连接服务端数据库，并对数据库进行增删改减，复习强化了 sql 语言的应用，到从服务器端实现对 API 的参数传递和调用，无不是收获颇丰，受益匪浅。

一个月期间有很多次不知所措，体会了在寻找了各种办法之后还依然无法解决的问题的痛苦。从环境配置到地图发布，从服务端功能逻辑实现到前端结果显示，从功能堆砌到界面布局

优化设计，从页面跳转繁琐复杂到显示简洁综合，遇到了很多很多大家共性的问题，也遇到了很多自己独特的问题在坚持不懈寻找答案的同时，也在不断地发现变换思路的重要性和带来的意想不到的效果，也发现和老师同学多多交流真的可以节省很多无用功，相互之间可能存在相同的问题，及时的沟通交流使得解决问题的办法更加简洁优化，提高效率，节省时间。同时资料的查找能力也是我们应该不断提高和总结的，如何高效的获取最有用的信息，找到解决方案也是我们需要学习的重要一环。